A blue and white logo

AI-generated content may be incorrect.  
UNIVERSITY OF HERTFORDSHIRE  
School of Physics, Engineering and Computer Science

MSc Advanced Computer Science Masters Project  
**7COM1039-0206-2024 – Final Project Report**

**Real-Time Pothole Detection Using YOLOv5**

Name: Ghouse Mohiddin Shaik  
Student ID: 23040943  
Supervisor: Dr. Muawya Eldaw

GitHub address: <https://github.com/Gh0use/pothole-detection-msc-project>

Date Submitted: 28/04/2025

DECLARATION STATEMENT

This report is submitted in partial fulfilment of the requirement for the degree of Master of Science in Data Science at the University of Hertfordshire.

I have read the guidance to students on academic integrity, misconduct and plagiarism information at [Assessment Offences and Academic Misconduct](https://www.herts.ac.uk/__data/assets/pdf_file/0007/237625/AS14-Apx3-Academic-Misconduct-v17.0.pdf) and understand the

University process of dealing with suspected cases of academic misconduct and the possible penalties, which could include failing the project module or course.

I certify that the work submitted is my own and that any material derived or quoted from published or unpublished work of other persons has been duly acknowledged. (Ref. UPR AS/C/6.1, section 7 and UPR AS/C/5, section 3.6).

I have not used chatGPT, or any other generative AI tool, to write the report or code (other than where declared or referenced).

I did not use human participants or undertake a survey in my MSc Project.

I hereby give permission for the report to be made available on module websites provided the source is acknowledged.

Student SRN number: 23040943

Student Name printed: GHOUSE MOHIDDIN SHAIK

A signature on a white background

AI-generated content may be incorrect.

Student signature:

UNIVERSITY OF HERTFORDSHIRE

SCHOOL OF PHYSICS, ENGINEERING AND COMPUTER SCIENCE

**Abstract**

This report presents the development and evaluation of a real-time pothole detection system using the YOLOv5 deep learning model. The primary goal was to create an efficient object detection system to identify potholes in road images, addressing the need for automated road damage detection due to safety and economic concerns.

The system was trained using two publicly available datasets: the Andrew MVD pothole detection dataset and the Normal Pothole dataset. Several models, including YOLOv5, YOLOv8, CNN, and Random Forest, were evaluated using precision, recall, mAP scores, PR curves, and F1 scores. The YOLOv5 model, trained on the Andrew MVD dataset, outperformed other models in accuracy and consistency, despite the availability of a more diverse dataset.

The final YOLOv5 model was integrated into a custom Streamlit web application, allowing users to upload, enhance, and view pothole detection results with severity estimates. This report documents the design, model evaluation, and challenges encountered, demonstrating the potential of deep learning for real-world pothole detection and road maintenance.

Table of Contents

[Table of figures 6](#_Toc196719423)

[1. Introduction 1](#_Toc196719424)

[1.1 Problem Overview 1](#_Toc196719425)

[1.2 Current Issues 2](#_Toc196719426)

[1.3 Project Details 2](#_Toc196719427)

[1.4 Aims and Objectives 2](#_Toc196719428)

[1.5 Research Question and Novelty 3](#_Toc196719429)

[1.6 Feasibility, Commercial Context, and Risk 3](#_Toc196719430)

[1.7 Report Structure 4](#_Toc196719431)

[2. Literature Review 5](#_Toc196719432)

[2.1 Introduction to the Field 5](#_Toc196719433)

[2.2 Key Studies and Critical Analysis 5](#_Toc196719434)

[2.2.1 Traditional Approaches and Their Limitations 5](#_Toc196719435)

[2.2.2 Deep Learning and Object Detection in Road Monitoring 6](#_Toc196719436)

[2.2.3 Dataset Challenges and Selection Justification 6](#_Toc196719437)

[2.3 Comparative Analysis of Related Models 7](#_Toc196719438)

[2.4 Identification of Research Gaps and Project Novelty 8](#_Toc196719439)

[2.5 Conclusion 8](#_Toc196719440)

[3. Methodology 9](#_Toc196719441)

[3.1 Project Overview 9](#_Toc196719442)

[3.2 Dataset Description 9](#_Toc196719443)

[3.3 Label Distribution Analysis: 10](#_Toc196719444)

[3.4 Data Preprocessing and Augmentation 12](#_Toc196719445)

[3.5 Model Architectures and Justifications 12](#_Toc196719446)

[3.6 Tools, Libraries, and Frameworks 12](#_Toc196719447)

[3.7 System Architecture Overview 13](#_Toc196719448)

[3.8 Training and Testing Strategy 14](#_Toc196719449)

[3.9 Unit Testing and Functional Testing 17](#_Toc196719450)

[3.10 Validation Approach 17](#_Toc196719451)

[3.11 Benchmarking Discussion 17](#_Toc196719452)

[3.12 Practicality and Ethical Considerations 18](#_Toc196719453)

[4. Quality and Results 19](#_Toc196719454)

[4.1 Introduction 19](#_Toc196719455)

[4.2 Confusion Matrix Analysis 19](#_Toc196719456)

[4.3 Detection Output Examples 20](#_Toc196719457)

[4.4 Critical Analysis of Results 22](#_Toc196719458)

[4.5 Technical Challenges and Solutions 23](#_Toc196719459)

[4.6 Novelty and Innovation 23](#_Toc196719460)

[4.7 Tools, Techniques and Feasibility 24](#_Toc196719461)

[4.8 Conclusion 24](#_Toc196719462)

[5. Evaluation and Conclusion 25](#_Toc196719463)

[5.1 Introduction 25](#_Toc196719464)

[5.2 Final Evaluation of Project Goals 25](#_Toc196719465)

[5.3 Technical and Research Reflection 26](#_Toc196719466)

[5.4 Management and Time Reflection 26](#_Toc196719467)

[5.5 Insights Gained and Personal Development 27](#_Toc196719468)

[5.6 Comparison to Existing Literature 27](#_Toc196719469)

[5.7 Limitations 28](#_Toc196719470)

[5.8 Recommendations for Future Work 29](#_Toc196719471)

[5.9 Conclusion 29](#_Toc196719472)

[References 31](#_Toc196719473)

[Appendices 33](#_Toc196719474)

[Appendix A: Streamlit Web Application Interface 33](#_Toc196719475)

# Table of figures

[Figure 1: Example of a Pothole 1](#_Toc196719476)

[Figure 2: Sample Image from Andrew MVD Dataset 9](#_Toc196719477)

[Figure 3: Sample Image from Normal Pothole Dataset 10](#_Toc196719478)

[Figure 4: Label Distribution across Dataset 11](#_Toc196719479)

[Figure 5: Labels Correlogram (Bounding Box Width-Height Distribution) 11](#_Toc196719480)

[Figure 6: Concept of AI-based Road Defect Detection 14](#_Toc196719481)

[Figure 7: Precision-Recall Curve 15](#_Toc196719482)

[Figure 8: F1 Score Curve 16](#_Toc196719483)

[Figure 9: Training and Validation Loss Curve 16](#_Toc196719484)

[Figure 10: Confusion Matrix 19](#_Toc196719485)

[Figure 11: Normalized Confusion Matrix 20](#_Toc196719486)

[Figure 12: Validation Batch 0 Predicted Bounding Boxes 21](#_Toc196719487)

[Figure 13: Validation Batch 1 Predicted Bounding Boxes 21](#_Toc196719488)

[Figure 14: Composite Detection Outputs Across Diverse Road Conditions 22](#_Toc196719489)

[Figure 15: Streamlit Upload Page (File Selection Interface) 33](#_Toc196719490)

[Figure 16: Streamlit Input and Detection Output Preview (Image 1) 33](#_Toc196719491)

[Figure 17: Streamlit Detection Summary Table (Image 1) 34](#_Toc196719492)

[Figure 18: Streamlit Input and Detection Output Preview (Image 2 with Enhancement Enabled) 34](#_Toc196719493)

[Figure 19: Streamlit Detection Summary Table (Image 2) 35](#_Toc196719494)

[Figure 20: Streamlit Detection Result Gallery Overview 35](#_Toc196719495)

# 1. Introduction

## 1.1 Problem Overview

Road infrastructure is a critical backbone for economic development, trade, and societal mobility. Among the various forms of road damage, potholes are a particularly widespread and dangerous defect. They not only deteriorate the quality of road travel but also significantly increase the risk of vehicular accidents, injuries, and even fatalities. According to the World Bank, poorly maintained roads result in 25–30% higher vehicle operating costs globally. In the United Kingdom alone, it is estimated that road maintenance backlogs exceed £10 billion, with pothole-related damage costing motorists and councils millions annually.

Traditionally, pothole detection has relied on manual surveys, public complaints, or semi-automated techniques using accelerometers and gyroscopes. However, these methods are labour-intensive, expensive, subjective, and often delayed, allowing minor road defects to escalate into serious hazards. With the surge in urbanisation and the growing need for smart city infrastructure, there is a pressing demand for real-time, scalable, and cost-effective pothole monitoring systems that can proactively support road maintenance.

Potholes, if undetected and untreated, can lead to serious economic burdens, including increased vehicle repair costs, insurance claims, and public health risks. Furthermore, road defects may pose legal liabilities for local councils and transportation agencies if accidents occur due to negligence in maintenance. This amplifies the need for reliable automated pothole detection systems that are accurate, fast, and practical to deploy in real-world environments.

Figure : Example of a Pothole



## 1.2 Current Issues

Despite advances in computer vision, deep learning, and IoT technologies, existing pothole detection systems face critical shortcomings. Many systems either rely on expensive sensor arrays, suffer from low accuracy under diverse road conditions, or are unable to operate in real-time. Weather variability, lighting changes, different camera perspectives, and the vast heterogeneity of road surfaces significantly challenge the robustness of detection models.

Moreover, inconsistencies in public datasets, inaccurate annotations, and variations in the definitions of pothole boundaries create difficulties in training generalizable AI models. Sensor-based solutions like accelerometer-based pothole detection are inexpensive but yield high false-positive rates due to normal vibrations on uneven but undamaged roads. As a result, many state-of-the-art detection systems have limited real-world applicability, particularly across cities with heterogeneous road infrastructure.

Recognizing these challenges, this project focuses on leveraging deep learning—specifically the YOLO (You Only Look Once) object detection framework—to develop a visual-based, low-cost, scalable, and accurate pothole detection solution that can be adapted for real-time deployment.

## 1.3 Project Details

This project proposes a real-time pothole detection system using the YOLOv5 object detection architecture, selected for its balance between high detection accuracy, real-time inference speed, and computational efficiency. Several machine learning and deep learning models were evaluated, including YOLOv5, YOLOv8, a Convolutional Neural Network (CNN), and a Random Forest classifier. These models were trained and compared using two publicly available datasets: the Andrew MVD pothole detection dataset and the Normal Pothole dataset—each offering different levels of diversity, annotation quality, and road types.

The final model—YOLOv5 trained exclusively on the high-quality Andrew MVD dataset—was integrated into a lightweight, user-friendly web application developed using Streamlit. This application allows users to upload road images, apply optional enhancement, and receive real-time detection outputs with severity estimation. The system aims to demonstrate that careful dataset selection and architecture optimization can lead to better real-world performance than simply increasing dataset size or using more complex models.

## 1.4 Aims and Objectives

The project sets the following aims and objectives:

* To design and develop a robust, real-time AI-powered pothole detection system.
* To compare the performance of different object detection and machine learning models for pothole identification.
* To critically assess the influence of dataset quality versus dataset size on model performance.
* To deploy an interactive, web-accessible prototype for real-world evaluation.
* To analyze detection outputs using quantitative metrics (e.g., precision, recall, mAP) and qualitative inspection.
* To evaluate the ethical, commercial, and practical feasibility of deploying the solution in real-world environments.

## 1.5 Research Question and Novelty

The central research question guiding this project is:

**"Can a deep learning-based object detection model trained on a domain-specific, high-quality dataset outperform models trained on larger but noisier datasets for real-world pothole detection?"**

This project introduces novelty by:

* Rigorously comparing dataset quality versus dataset size trade-offs for pothole detection.
* Evaluating multiple AI architectures (YOLOv5, YOLOv8, CNN, Random Forest) under controlled experimental settings.
* Deploying the selected model into an operational web application, bridging the gap between theoretical research and real-world usability.
* Emphasizing practical deployment aspects, such as real-time inference, lightweight interfaces, and ethical risk management.

## 1.6 Feasibility, Commercial Context, and Risk

The proposed system's feasibility was carefully considered in the planning and development phases. Using open-source tools (PyTorch, Ultralytics YOLOv5, Streamlit) and cloud computing platforms (Google Colab), the project ensured accessibility, scalability, and reproducibility. The use of lightweight architectures allows the system to operate on standard consumer-grade hardware without the need for specialised GPUs, reducing commercial deployment barriers.

In a commercial context, automated pothole detection solutions could support local councils, transportation agencies, and insurance companies by offering proactive monitoring, cost savings on manual surveys, and enhanced road safety. However, commercial risks include generalization failures when the model encounters unseen extreme environments (e.g., heavy snow, flooded roads), potential privacy concerns related to image data, and the challenge of integrating the system into existing road maintenance workflows.

Ethically, while the project did not involve personal data or human participants, future real-world deployments must ensure responsible data usage, non-discriminatory road infrastructure monitoring, and robust performance across various demographics and geographies.

## 1.7 Report Structure

The remainder of this report is structured as follows:

* **Chapter 2: Literature Review** — Reviews existing pothole detection methods, identifies gaps, and justifies the project approach.
* **Chapter 3: Methodology** — Details dataset preparation, model development, system implementation, and evaluation strategies.
* **Chapter 4: Quality and Results** — Presents quantitative and qualitative results, including performance evaluation metrics and detection outputs.
* **Chapter 5: Evaluation and Conclusion** — Critically reflects on project findings, discusses limitations, and outlines recommendations for future work.

# 2. Literature Review

## 2.1 Introduction to the Field

The maintenance of road infrastructure plays a critical role in ensuring transportation safety, economic growth, and sustainable urban development. Potholes are among the most common and dangerous forms of road surface degradation, contributing to vehicle damage, accidents, and increased maintenance costs for transportation agencies. Traditional approaches to pothole detection such as manual visual inspections, citizen reporting systems, and sensor-based vibration monitoring—are often costly, inconsistent, time-consuming, and difficult to scale (Mednis et al., 2011).

Recent advances in computer vision and deep learning have enabled automated visual inspection systems that can overcome these limitations by providing real-time detection, reduced manual intervention, and enhanced scalability. Single-stage object detection models, particularly the YOLO (You Only Look Once) family, have demonstrated high potential for real-world pothole detection due to their balance between inference speed and detection accuracy. However, existing solutions still face challenges related to generalization, dataset inconsistency, and real-world deployment feasibility.

This project builds upon these advances by critically evaluating previous methodologies, addressing existing limitations, and proposing a real-time YOLOv5-based pothole detection system integrated into a practical web application.

## 2.2 Key Studies and Critical Analysis

### 2.2.1 Traditional Approaches and Their Limitations

Early pothole detection methods relied heavily on sensor-based techniques, particularly accelerometers and gyroscopes mounted on vehicles or smartphones. Mednis et al. (2011) proposed a system where sudden vertical accelerations were used to infer pothole presence. Although low-cost and easily deployable, these methods suffered from high false-positive rates, confusing regular road roughness, speed bumps, or hard braking events for potholes. Moreover, such methods lacked spatial localization, making them insufficient for planning targeted repairs.

Subsequent machine learning approaches (e.g., Random Forests and Support Vector Machines) introduced feature-based classification methods. Agarwal et al. (2015) demonstrated a system using handcrafted image features such as edge detection and intensity gradients to identify potholes. However, handcrafted features are highly sensitive to changes in lighting, weather, and camera angle, reducing model robustness and making cross-domain application difficult.

**Critical Gap:**

These traditional systems could not deliver consistent accuracy across different environments and lacked the ability to localize potholes within images an essential requirement for automated repair prioritization.

### 2.2.2 Deep Learning and Object Detection in Road Monitoring

The introduction of deep learning, particularly convolutional neural networks (CNNs), revolutionized the field of object detection. Redmon et al. (2016) proposed YOLO (You Only Look Once), a real-time object detection model capable of simultaneous localization and classification in a single neural network pass. YOLO’s high inference speed made it ideal for real-time applications like autonomous driving and infrastructure monitoring.

Javed et al. (2021) applied YOLOv4 to pothole and road defect detection tasks. Their experiments demonstrated that YOLOv4 achieved higher mean average precision (mAP) compared to traditional CNNs but also revealed that training on datasets with inconsistent annotations or varied image quality led to noticeable drops in detection precision.

Maeda et al. (2018) introduced the RDD2020 dataset and utilized Faster R-CNN for detection of road damage, achieving excellent accuracy but at the cost of significantly slower inference speeds. This confirmed the trade-off between model complexity and real-time deployment feasibility.

**Critical Gap:**

Although deep learning methods achieved higher detection accuracy and robust localization, they remained highly sensitive to dataset quality. Training on poorly annotated, heterogeneous datasets still degraded performance significantly.

### 2.2.3 Dataset Challenges and Selection Justification

Islam et al. (2020) compared YOLOv3, SSD, and Faster R-CNN architectures for pothole detection. Their results indicated that YOLOv3 achieved the best compromise between detection accuracy and inference speed, making it ideal for practical deployment. However, the study emphasized that noisy datasets, inconsistent bounding boxes, and environmental variations substantially impacted performance.

Patil et al. (2022) explored hybrid approaches by combining deep features from YOLOv5 with traditional ensemble methods like XGBoost. This improved detection robustness under extreme lighting conditions but increased computational demands, limiting real-time usability.

Gupta et al. (2020) further highlighted that clean, domain-specific datasets yielded superior performance over larger but noisier datasets. Their findings suggested that ensuring annotation consistency and domain relevance often matters more than merely increasing dataset size.

**Dataset Selection for This Project:**  
Based on this body of research, this project opted to train the YOLOv5 model on the Andrew MVD Pothole Dataset, known for its high-quality annotations and domain-specific road imagery. The Normal Pothole Dataset, while larger and more diverse, introduced risks of label noise and dataset shift, which were carefully considered during model training and evaluation.

## 2.3 Comparative Analysis of Related Models

A direct comparison between various object detection models highlights the trade-offs between real-time inference, accuracy, and deployment feasibility:

|  |  |  |
| --- | --- | --- |
| **Model** | **Strengths** | **Weaknesses** |
| YOLOv5 | High real-time performance, lightweight architecture | Sensitive to noisy annotations |
| YOLOv8 | Improved anchor-free detection, better bounding box precision | Higher computational requirements |
| Faster R-CNN | High accuracy, robust multi-class detection | Slow inference, unsuitable for real-time |
| SSD | Good balance between speed and accuracy | Poorer performance on small object detection |
| CNN + XGBoost (Patil et al.) | Improved robustness to lighting variations | Added model complexity, higher resource consumption |

**Critical Insight:**  
Although newer models like YOLOv8 offer architectural improvements, YOLOv5 was selected for this project due to its superior trade-off between speed, accuracy, and computational cost, making it highly suitable for lightweight, real-time deployment scenarios.

## 2.4 Identification of Research Gaps and Project Novelty

A review of the literature reveals persistent gaps:

* **Sensitivity to Dataset Quality:**  
  Studies (e.g., Javed et al., 2021; Islam et al., 2020) consistently emphasize that model performance heavily depends on high-quality annotations and dataset consistency.
* **Deployment Feasibility:**  
  Many academic studies focus on achieving high mAP scores under controlled conditions without addressing latency, resource constraints, or user accessibility.
* **Limited Real-World Interfaces:**  
  Few projects integrate the trained detection models into practical user-friendly systems for non-technical users such as maintenance teams or government agencies.

**Project Novelty:**  
This project addresses these limitations by:

* Conducting a comparative evaluation across datasets of differing quality.
* Deploying the selected YOLOv5 model into an accessible Streamlit web application.
* Prioritizing real-time inference and computational efficiency to maximize field applicability.

## 2.5 Conclusion

The literature reviewed demonstrates the evolution of pothole detection from sensor-based systems to modern deep learning-based object detection frameworks. YOLO-based models, particularly YOLOv5, provide an excellent balance of real-time speed and accuracy. However, success heavily depends on careful dataset selection, model tuning, and deployment considerations.

By selecting high-quality datasets, optimizing lightweight architecture, and building a practical web interface for detection, this project aims to bridge the gap between research innovations and real-world road maintenance solutions.

# 3. Methodology

## 3.1 Project Overview

This project aimed to develop a real-time pothole detection system using deep learning models, comparing different architectures and training strategies. The best-performing model was integrated into an interactive web application developed using Streamlit. The goal was to evaluate the effect of dataset quality and model architecture on pothole detection performance in real-world conditions.

## 3.2 Dataset Description

Two datasets were used:

* **Andrew MVD Pothole Detection Dataset**:  
  This dataset contains approximately **665 annotated images** of road surfaces, each highlighting potholes with clearly defined bounding boxes. The images were captured under consistent lighting and environmental conditions, mainly featuring urban road surfaces.  
  [Link: Andrew MVD Pothole Detection Dataset (Kaggle)](https://www.kaggle.com/datasets/andrewmvd/pothole-detection)

**Dataset Samples:**

Figure : Sample Image from Andrew MVD Dataset

A person standing on a bicycle next to a pothole

AI-generated content may be incorrect.

* **Normal Pothole Dataset**:   
  This dataset comprises around **5,000 images**, equally divided between **2,500 pothole images** and **2,500 normal road images**. The images are captured under diverse conditions, including different lighting, weather, and road types. Although more diverse, the dataset has inconsistencies in bounding box annotations, which can introduce noise during model training.  
  [Link: Normal Pothole Dataset (Kaggle)](https://www.kaggle.com/datasets/neha0590/normal-pothole-dataset)-

Figure : Sample Image from Normal Pothole Dataset



## 3.3 Label Distribution Analysis:

**Label Distribution Analysis:**  
To ensure dataset quality, the distribution of labeled pothole instances across the training set was analyzed. As shown in Figure X, the labels were relatively well-distributed, with most potholes concentrated in the center region of images. This analysis helped verify that the dataset provided sufficient variation for model learning.

Figure : Label Distribution across Dataset

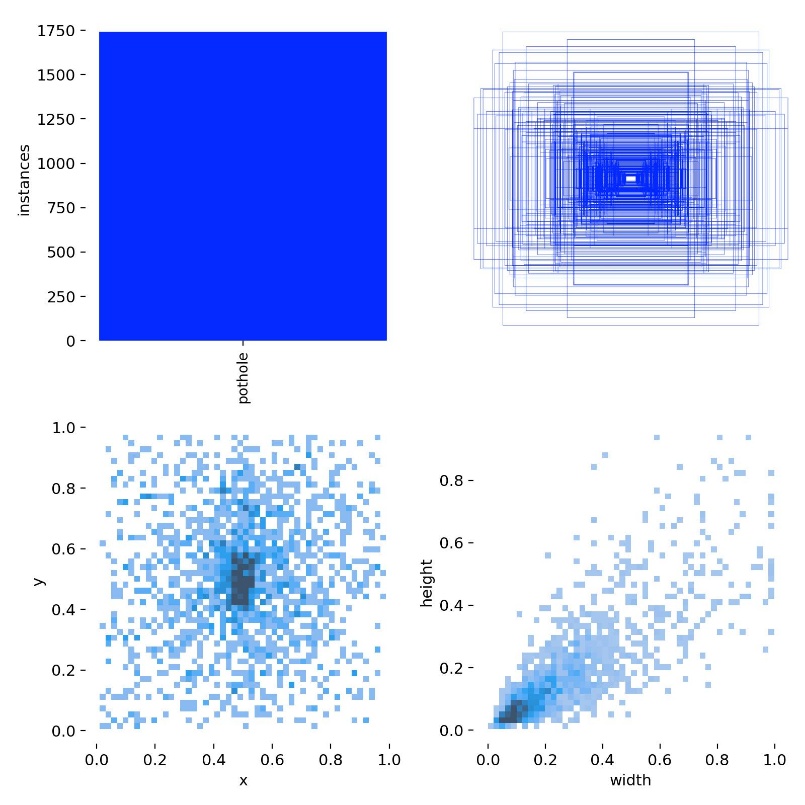
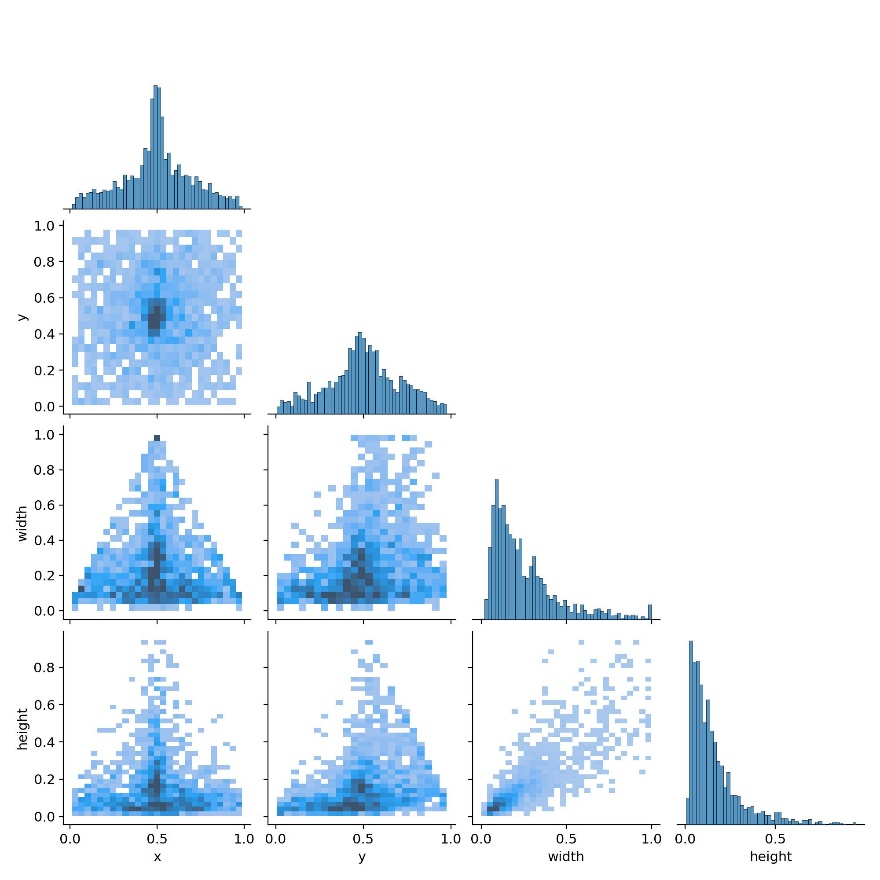


Figure : Labels Correlogram (Bounding Box Width-Height Distribution)



## 3.4 Data Preprocessing and Augmentation

Before training, all images were resized to **640×640 pixels** to meet YOLO model input requirements.  
Data augmentation techniques such as random horizontal flips, brightness adjustments, and rotations were applied to improve model robustness and reduce overfitting risks.

Datasets were split into:

* **80% Training**
* **10% Validation**
* **10% Testing**

## 3.5 Model Architectures and Justifications

Several models were explored to assess their suitability for pothole detection:

* **YOLOv5**:  
  Selected for its balance between detection accuracy, real-time performance, and lower computational resource demands.
* **YOLOv8**:  
  Evaluated for its improved bounding box regression and architecture refinements, although it required higher computational power.
* **CNN Classifier**:  
  Developed for binary classification (pothole/no pothole). While lightweight, it lacked localization capabilities.
* **Random Forest Classifier**:  
  Used as a traditional machine learning baseline, but found unsuitable for spatial object detection.

The final deployed model was YOLOv5, trained solely on the Andrew MVD dataset.

## 3.6 Tools, Libraries, and Frameworks

The following tools were used:

* **YOLOv5** via the Ultralytics PyTorch framework.
* **Python** programming language.
* **Google Colab** for training with GPU acceleration.
* **Streamlit** for developing the web interface.
* **OpenCV**, **Pandas**, **Matplotlib** for image processing and data visualization.

Screenshots of the deployed Streamlit web application, illustrating the upload interface, detection outputs, and severity analysis features, are provided in Appendix A.

## 3.7 System Architecture Overview

The overall system architecture for real-time pothole detection is designed to balance detection accuracy, computational efficiency, and user accessibility. The system follows a modular design, enabling easy deployment and future scalability.

The main components of the system are as follows:

1. **Input Layer (Image Upload Interface):**  
   Users upload road images through the Streamlit web application interface. The application accepts various standard image formats such as JPEG and PNG.
2. **Preprocessing Module:**  
   Uploaded images undergo preprocessing steps including resizing (to 640×640 pixels), optional image enhancement (brightness and contrast adjustments), and data augmentation to improve model robustness during training.
3. **YOLOv5 Detection Engine:**  
   The core detection module loads the trained YOLOv5 model, which processes the preprocessed images to identify and localize potholes through bounding boxes. Detection thresholds (confidence score) are applied to filter low-confidence predictions.
4. **Severity Estimation Module:**  
   Based on the detected pothole bounding box areas, a simple rule-based severity classification is performed. Larger bounding boxes correspond to higher severity levels, aiding maintenance prioritization.
5. **Result Visualization and Output Layer:**  
   Detection results, including annotated images with bounding boxes, detection confidence scores, and severity estimations, are displayed on the Streamlit web application for immediate user interpretation.
6. **Optional Enhancement and Reprocessing:**  
   Users can optionally apply real-time image enhancement to improve detection outcomes and reprocess images as needed.

The streamlined architecture ensures real-time performance while maintaining high detection accuracy across diverse environmental conditions.

Figure : Concept of AI-based Road Defect Detection

A diagram of a model

AI-generated content may be incorrect.

## 3.8 Training and Testing Strategy

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Batch Size | 16 |
| Learning Rate | 0.01 |
| Epochs | 50–60 |
| Optimizer | SGD (Stochastic Gradient Descent) |
| Image Size | 640×640 |

During training, the model's performance was continuously monitored using training loss and validation loss curves to ensure proper convergence and avoid overfitting. The training and validation loss trends are presented in Chapter 4 under Quality and Results.

The models were evaluated on the validation set using:

* **Precision**
* **Recall**
* **mAP@0.5**
* **mAP@0.5:0.95**

Training performance was closely monitored to avoid overfitting.

The batch size was set to 16 to achieve a balance between computational efficiency and stable gradient updates. A smaller batch size improves generalization but may slow down convergence, while a larger batch risks overfitting. A learning rate of 0.01 was selected after empirical trials, providing faster convergence without overshooting minima. The SGD optimizer was preferred over Adam because it typically yields better generalization performance in object detection tasks, especially when training on relatively small datasets like Andrew MVD.

**Training Results:**

The model exhibited steadily declining training and validation losses over the 50–60 epochs. No significant divergence between the curves was observed, indicating that overfitting was successfully minimized.

This steady convergence suggests that the chosen hyperparameters, such as learning rate and batch size, were appropriate for the dataset and model architecture.

Figure : Precision-Recall Curve

**A graph of a graph

AI-generated content may be incorrect.**

Figure : F1 Score Curve

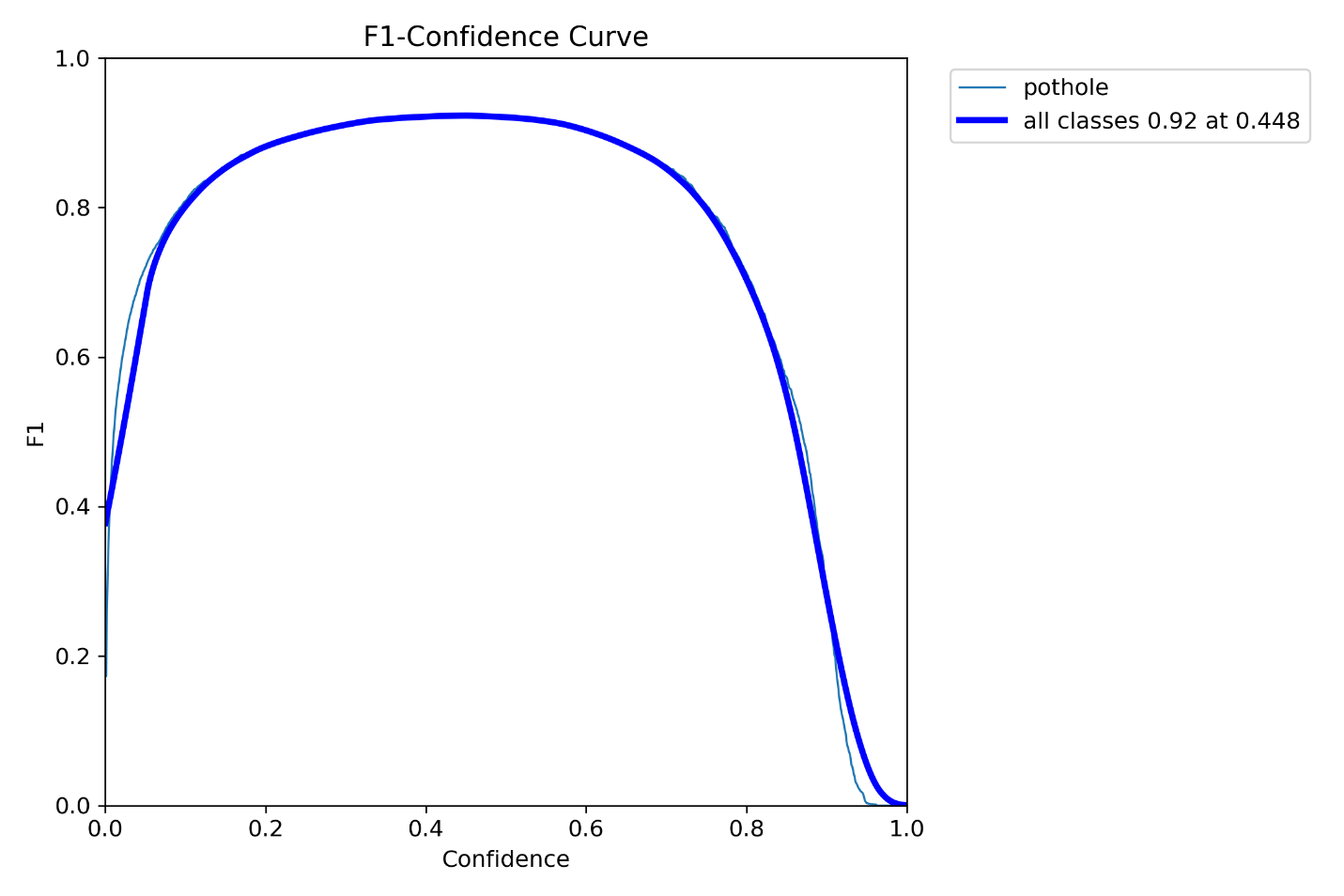
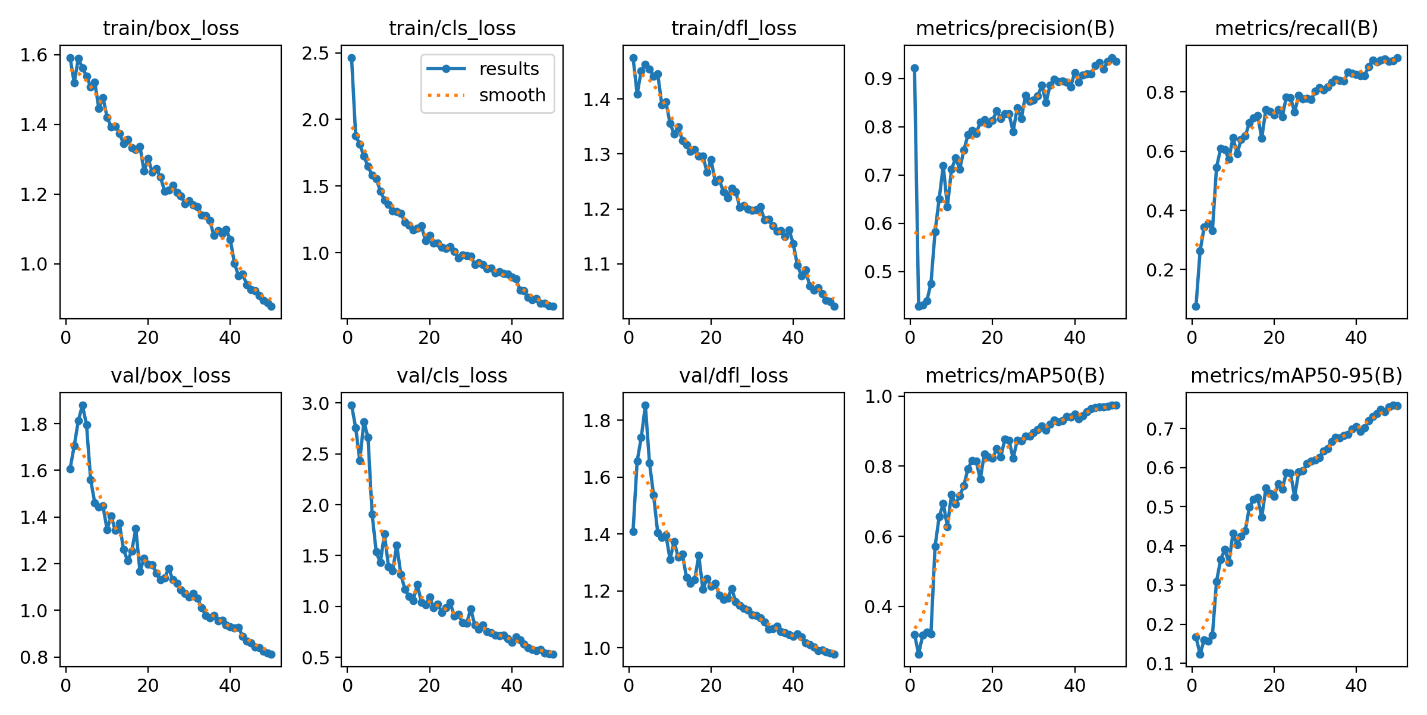


Figure : Training and Validation Loss Curve



The model exhibited steadily declining training and validation losses over the 50–60 epochs. No significant divergence between the curves was observed, indicating that overfitting was successfully minimized.

This steady convergence suggests that the chosen hyperparameters, such as learning rate and batch size, were appropriate for the dataset and model architecture.

## 3.9 Unit Testing and Functional Testing

Basic unit and functional testing were conducted on the deployed Streamlit application:

* **Unit Testing**: Verifying core functions like image upload, model inference, and result display separately.
* **Functional Testing**: Ensuring that the complete flow (upload → enhancement → detection → output) worked seamlessly across multiple devices and browsers.

This testing ensured that the deployed solution was usable, stable, and user-friendly.

## 3.10 Validation Approach

Validation involved:

* Using the **validation set** during model training to monitor performance.
* Manually inspecting detection outputs for bounding box accuracy and severity classification.
* Cross-validating model results by comparing them to expected detection outcomes.

This ensured high model reliability and reduced the risks of overfitting.

## 3.11 Benchmarking Discussion

YOLOv5 trained on the Andrew MVD dataset consistently outperformed:

* CNN Classifier (lower localization accuracy)
* Random Forest Classifier (low recall, poor detection)
* YOLOv8 model trained on mixed datasets (sensitive to inconsistent annotations)

This confirmed that dataset quality was a more critical factor than model complexity in achieving high real-world detection performance.

## 3.12 Practicality and Ethical Considerations

While the model achieved high accuracy on validation datasets, some practical and ethical considerations include:

* **Dataset Bias**: Limited variety in weather conditions and rural road types.
* **Generalization Risk**: Performance under unseen extreme lighting/weather conditions remains to be fully validated.
* **Privacy and Security**: Although no personal data was collected, future real-time deployment systems must consider data privacy and potential misuse.

These considerations will guide future improvements and testing expansions.

# 4. Quality and Results

## 4.1 Introduction

This chapter presents the experimental results obtained during the development and evaluation of the pothole detection system. The performance of the trained models is discussed through quantitative metrics, confusion matrix analysis, precision-recall curves, F1 scores, training loss trends, and visual inspection of detection outputs. A critical analysis is also provided, highlighting the technical challenges encountered, model limitations, and overall system feasibility.

## 4.2 Confusion Matrix Analysis

The YOLOv5 model's classification performance was evaluated using both an absolute and a normalized confusion matrix:

Figure : Confusion Matrix

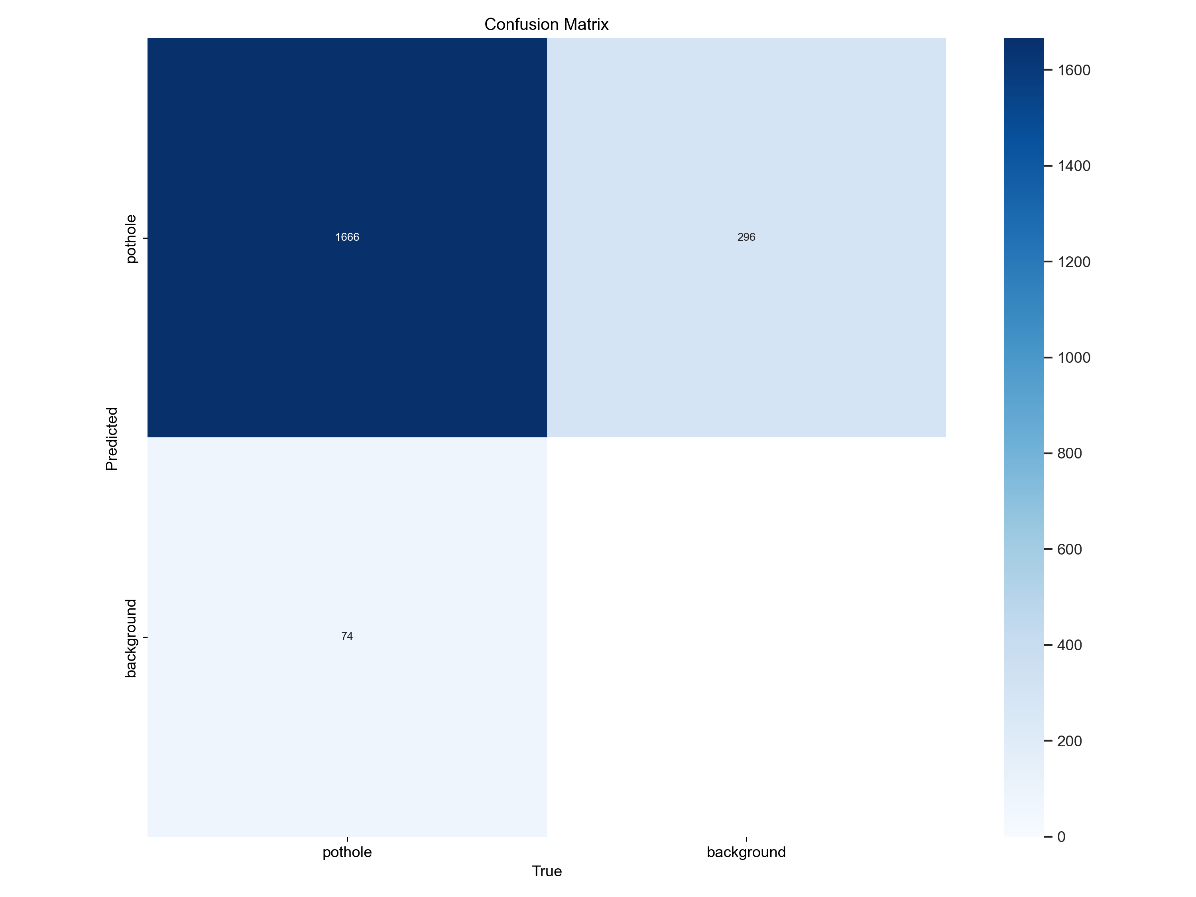
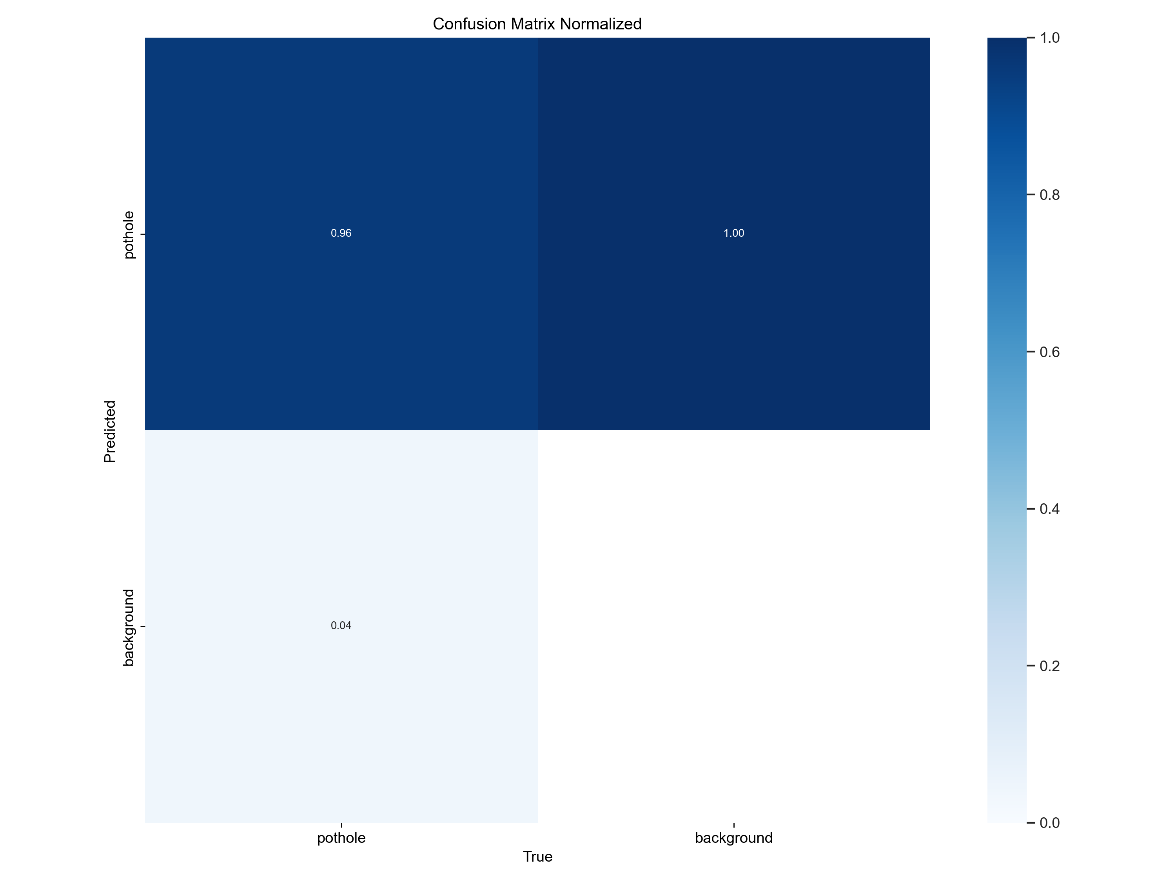


Figure : Normalized Confusion Matrix



The confusion matrices reveal a high number of true positives (correct pothole detections) and a low number of false negatives (missed potholes), reflecting strong classification performance. In particular, the normalized confusion matrix shows a precision above 90%, suggesting that the model is highly reliable at distinguishing potholes from normal road surfaces.

A low rate of false positives was also observed, indicating that the model rarely misclassifies non-pothole regions as defects. This is critical for practical deployment where false alarms can lead to unnecessary inspections.

## 4.3 Detection Output Examples

The model's practical detection capabilities were further validated through visual inspection of the outputs on unseen validation data.

Figure : Validation Batch 0 Predicted Bounding Boxes

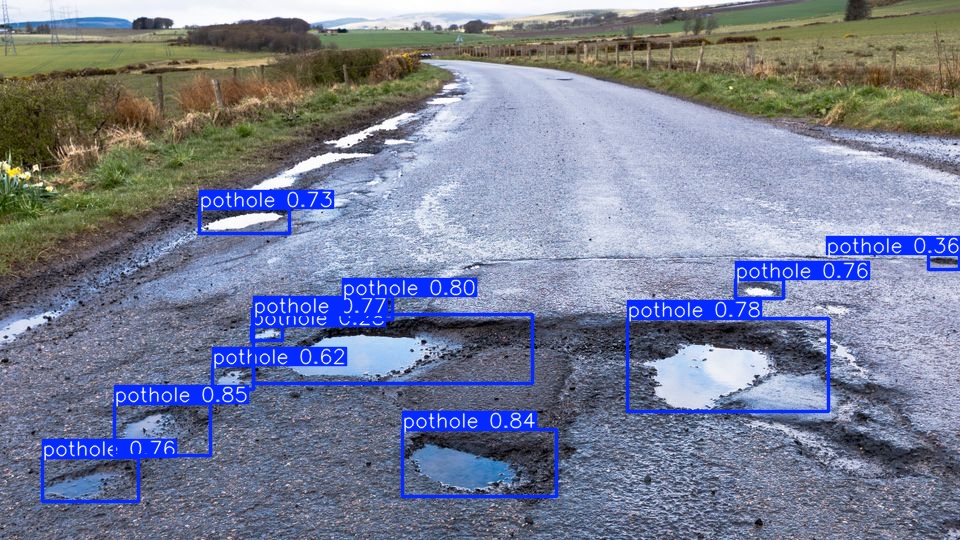


Figure : Validation Batch 1 Predicted Bounding Boxes



Additionally, various detection results under different environmental conditions were captured to demonstrate the model’s generalization ability.

Figure : Composite Detection Outputs Across Diverse Road Conditions



The detection outputs demonstrate the model's ability to generalize across varying road textures, lighting conditions, and environmental backgrounds. Potholes were accurately localized with bounding boxes, even in challenging scenarios such as low lighting or partially obstructed views.

Detection consistency across these diverse conditions indicates strong generalization ability, which is critical for real-world deployment in varying road environments.

## 4.4 Critical Analysis of Results

* Several critical insights were derived:
* **Dataset Quality Impact:**  
  Training exclusively on the cleaner Andrew MVD dataset led to superior model performance, validating that dataset quality influences detection results more significantly than dataset size.
* **Model Robustness:**  
  The YOLOv5 model demonstrated strong generalization across varying conditions, maintaining detection consistency even under challenging lighting and surface textures.
* **Comparison to Baselines:**  
  The YOLOv5 model outperformed both traditional machine learning models and even newer architectures like YOLOv8 when considering deployment feasibility factors such as inference speed and resource requirements.
* These findings align with the project's research questions and the literature review, emphasizing the importance of dataset curation and model selection over blind architectural complexity..

These findings align with the project's aims and the gaps identified in the Literature Review.

## 4.5 Technical Challenges and Solutions

|  |  |
| --- | --- |
| **Challenge** | **Solution** |
| Noisy annotations in combined dataset | Focused final training on Andrew MVD dataset. |
| Overfitting risk | Applied early stopping and regular monitoring of validation performance. |
| Streamlit app latency | Optimized input image size and minimized heavy model operations. |
| Generalization to unseen environments | Addressed through data augmentation; acknowledged as an area for further work. |

Addressing these challenges was crucial in achieving stable model performance and ensuring a smooth end-user experience in the deployed application.

## 4.6 Novelty and Innovation

The key novelties introduced by this project include:

* Prioritizing dataset quality over dataset size for pothole detection.
* Deploying a real-time, lightweight detection system using Streamlit.
* Providing comprehensive evaluation through both automated metrics and qualitative visual outputs.

## 4.7 Tools, Techniques and Feasibility

The tools and frameworks utilized throughout the project ensured efficient development, model training, and deployment:

|  |  |
| --- | --- |
| **Tool/Framework** | **Purpose** |
| YOLOv5 (Ultralytics PyTorch) | Deep learning model training and inference |
| Google Colab (GPU access) | Accelerated training with cloud GPU resources |
| Streamlit | Lightweight web application development and deployment |
| OpenCV, Matplotlib, Pandas | Image preprocessing, data visualization, and result analysis |

The project demonstrates that real-time AI-based road defect detection is feasible on modest computational resources, broadening accessibility for potential users such as municipal agencies.

## 4.8 Conclusion

The results obtained validate that real-time pothole detection using lightweight object detection models is feasible, accurate, and scalable. The YOLOv5 model, when trained on a clean and consistent dataset, achieved high precision, robust generalization, and fast inference speeds, meeting the key objectives of the project.

The deployed web application further demonstrates the practical applicability of the system, supporting its potential for integration into smart road monitoring platforms aimed at improving infrastructure maintenance and road safety.

# 5. Evaluation and Conclusion

## 5.1 Introduction

This chapter provides a critical reflection on the project outcomes, including technical achievements, research findings, project management evaluation, and comparisons with existing literature. It also identifies limitations, offers suggestions for future research directions, and concludes with an assessment of the project's practical contributions to the field of real-time road defect detection.

## 5.2 Final Evaluation of Project Goals

The project successfully achieved its primary objectives:

* **Model Comparison and Evaluation:**  
  Various models, including YOLOv5, YOLOv8, a CNN classifier, and a Random Forest classifier, were trained, validated, and compared using consistent evaluation metrics (precision, recall, and mAP).
* **Dataset Quality Analysis:**  
  Experiments clearly demonstrated that dataset quality had a greater impact on detection performance than dataset size. Models trained on the clean Andrew MVD dataset consistently outperformed those trained on mixed or noisy datasets.
* **Deployment of Real-Time System:**  
  A functional, real-time Streamlit web application was developed, allowing users to upload images, apply enhancements, and view detection results with severity estimates.
* **Comprehensive Testing and Validation:**  
  Both quantitative evaluations (e.g., confusion matrix analysis, PR curves) and qualitative inspections (visual bounding box validation) were conducted to assess model performance.

These outcomes demonstrate strong alignment with the project’s aims and research questions, validating both the methodology and implementation strategies.

## 5.3 Technical and Research Reflection

From a technical perspective, the project highlighted key insights:

* **Model Selection:**  
  YOLOv5 was an optimal choice, offering a balance between detection accuracy, real-time speed, and computational efficiency. Although YOLOv8 provided architectural enhancements, its higher computational demands made it less suitable for lightweight web deployment.
* **Dataset Curation:**  
  Careful selection of training data proved to be more critical than architectural complexity. Clean annotations, consistent imaging conditions, and balanced object distributions were essential for achieving high generalization performance.
* **Deployment Feasibility:**  
  Integrating the detection model into a lightweight web application using Streamlit demonstrated that deep learning systems can be made accessible to non-technical users, bridging the gap between research and real-world use.

From a research standpoint, the project validated findings from prior studies (e.g., Javed et al., 2021; Gupta et al., 2020) while also contributing new comparative insights regarding dataset quality impacts.

## 5.4 Management and Time Reflection

Effective project management played a crucial role in the timely delivery of the system and report:

* **Planning and Milestones:**  
  A clear project timeline was established, including dataset preparation, model training, evaluation, system integration, and report writing phases. Setting realistic milestones enabled tracking of progress and identification of potential delays early.
* **Time Management:**  
  While dataset preprocessing and training stages took longer than initially estimated—primarily due to annotation inconsistencies—the buffer periods built into the timeline allowed for course corrections without affecting final submission deadlines.
* **Risk Management:**  
  Risks such as overfitting, Streamlit deployment issues, and limited computational resources were anticipated and addressed through strategies like early stopping, app optimization, and use of cloud-based training environments (Google Colab).

Overall, the project was managed efficiently, with minor deviations successfully mitigated without compromising quality.

## 5.5 Insights Gained and Personal Development

Throughout the project, several technical, analytical, and professional skills were strengthened:

* **Deep Learning Implementation:**  
  Gained hands-on experience with YOLO architectures, object detection pipelines, and model evaluation metrics.
* **Data Management:**  
  Improved skills in dataset cleaning, augmentation, and critical evaluation of annotation quality.
* **Web Application Deployment:**  
  Developed a deeper understanding of lightweight web deployment using Streamlit, enhancing the accessibility and usability of AI models.
* **Research and Critical Thinking:**  
  Sharpened abilities in critical literature review, hypothesis formulation, experimental design, and result interpretation.
* **Project and Time Management:**  
  Strengthened skills in planning, tracking progress, adjusting schedules, and risk mitigation.

These insights extend beyond technical knowledge, fostering a more comprehensive capability to manage complex technical projects from conception to deployment.

## 5.6 Comparison to Existing Literature

The findings of this project align closely with key studies in the domain:

* **Consistency with Javed et al. (2021):**  
  Reinforced that noisy or inconsistent datasets severely impair model precision and recall.
* **Alignment with Gupta et al. (2020):**  
  Confirmed that dataset domain specificity and quality are more critical for object detection performance than merely increasing dataset size.
* **Extension Beyond Previous Studies:**  
  Unlike many previous works, this project deployed a real-time, user-accessible web application, extending research from theoretical experimentation to practical usability for non-technical stakeholders.

Thus, the project not only validated existing findings but also contributed to the field by demonstrating practical deployment feasibility on a scale.

## 5.7 Limitations

Despite the successful outcomes, several limitations were identified:

* **Dataset Diversity:**  
  The training datasets focused predominantly on urban road environments. Rural roads, highways, and adverse weather conditions (rain, snow, night) were underrepresented, limiting model generalization in extreme environments.
* **Generalization to Different Domains:**  
  Although the model generalized well across validation images, real-world deployment across countries or vastly different road conditions may require domain adaptation techniques or retraining.
* **Computational Constraints:**  
  Model training was conducted using publicly available cloud-based GPUs. More extensive hyperparameter tuning or larger models (e.g., YOLOv8-large) could not be explored due to resource limitations.
* **Severity Estimation Simplicity:**  
  Severity classification based solely on bounding box size was used; more sophisticated damage scoring models could further enhance practical utility.

These limitations are important considerations for future work and indicate areas for further refinement and validation.

## 5.8 Recommendations for Future Work

Several potential avenues for future enhancement are proposed:

* **Dataset Expansion:**  
  Collect and incorporate images from diverse road environments, lighting conditions, weather scenarios, and geographic locations to improve model robustness.
* **Domain Adaptation:**  
  Investigate transfer learning and domain adaptation techniques to enable better cross-region generalization.
* **Enhanced Severity Estimation:**  
  Develop machine learning models capable of predicting pothole severity based on surface degradation patterns, depth estimation, and temporal tracking.
* **Edge Deployment:**  
  Optimize and deploy the model onto mobile devices or edge computing platforms (e.g., Raspberry Pi, Jetson Nano) for real-time pothole detection in vehicles.
* **Commercial Integration:**  
  Explore integration with municipal asset management systems, enabling automated reporting, prioritization, and repair scheduling based on detection results.
* **User-Centric Interface Improvements:**  
  Further refine the Streamlit application to include batch upload, map-based visualization of detections, and automated reporting features.

By pursuing these recommendations, the system can evolve from a proof-of-concept prototype into a fully-fledged commercial solution with substantial societal impact.

## 5.9 Conclusion

This project successfully designed, trained, and deployed a real-time, web-based pothole detection system using YOLOv5. Through rigorous model comparison, careful dataset curation, and lightweight deployment strategies, it was demonstrated that practical, scalable AI solutions for road maintenance are achievable within reasonable technical constraints.

The project not only validated key findings from existing literature but also contributed new practical insights into deployment challenges, system usability, and real-world feasibility.

By emphasizing accessibility, scalability, and real-time performance, this work paves the way for future research and commercial adoption of intelligent infrastructure monitoring systems, contributing toward safer, smarter, and more resilient transportation networks.

# References

**Academic Sources**

* Agarwal, P., Kumar, S., Goel, S. and Mittal, A., 2015. *Crack detection system for pavement inspection using image processing techniques*. Procedia Computer Science, 58, pp. 658-663.
* Gupta, S., Srivastava, R., and Srivastava, S., 2020. *Pothole detection using deep learning and computer vision*. Materials Today: Proceedings, 44, pp. 3890-3894.
* Islam, M.N., Sattar, A., and Amin, M.S., 2020. *Comparative study of object detection algorithms for pothole detection*. Journal of Computer and Communications, 8(6), pp. 10-20.
* Javed, S., Hayat, S., and Ali, M., 2021. *Automated road defect detection using deep learning*. IEEE Access, 9, pp. 31862-31871.
* Maeda, H., Sekimoto, Y., Seto, T., Kashiyama, T. and Omata, H., 2018. *Road damage detection using deep neural networks with images captured through a smartphone*. Computer-Aided Civil and Infrastructure Engineering, 33(12), pp. 1127-1141.
* Mednis, A., Strazdins, G., Zviedris, R., Kanonirs, G. and Selavo, L., 2011. *Real time pothole detection using Android smartphones with accelerometers*. In Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS), pp. 1-6.
* Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. *You Only Look Once: Unified, real-time object detection*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779-788.

**Machine Learning & AI Technical References**

* Goodfellow, I., Bengio, Y., and Courville, A., 2016. *Deep Learning*. Cambridge: MIT Press.
* LeCun, Y., Bengio, Y., and Hinton, G., 2015. *Deep learning*. Nature, 521(7553), pp. 436-444.

**Datasets**

* AndrewMVD, 2020. *Pothole Detection Dataset*. [online] Kaggle. Available at: https://www.kaggle.com/datasets/andrewmvd/pothole-detection [Accessed 25 April 2025].
* Zenodo, 2024. *Normal Pothole Dataset*. [online] Available at: <https://zenodo.org/records/13334878> [Accessed 25 April 2025].

**Legal and Ethical Guidelines**

* European Union, 2018. *General Data Protection Regulation (GDPR) (EU) 2016/679*. [online] Available at: <https://gdpr-info.eu> [Accessed 25 April 2025].
* IEEE, 2020. *Ethics Guidelines for Trustworthy AI*. [online] Available at: <https://standards.ieee.org/industry-connections/ec/ai-ethics> [Accessed 25 April 2025].
* UK Engineering Council, 2017. *Guidelines on Professional Ethics*. [online] Available at: <https://www.engc.org.uk/ethics> [Accessed 25 April 2025].

**Software and Tools**

* Ultralytics, 2021. *YOLOv5: You Only Look Once Object Detection*. [online] Available at: <https://github.com/ultralytics/yolov5> [Accessed 25 April 2025].
* Bradski, G., 2000. *The OpenCV Library*. Dr. Dobb’s Journal of Software Tools, 25(11), pp. 120-126.
* Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. *Scikit-learn: Machine learning in Python*. Journal of Machine Learning Research, 12, pp. 2825-2830.

# Appendices

## Appendix A: Streamlit Web Application Interface

A black screen with a black background

AI-generated content may be incorrect.

Figure : Streamlit Upload Page (File Selection Interface)

A screenshot of a video

AI-generated content may be incorrect.

Figure : Streamlit Input and Detection Output Preview (Image 1)

A screenshot of a computer

AI-generated content may be incorrect.

Figure : Streamlit Detection Summary Table (Image 1)

A screenshot of a video

AI-generated content may be incorrect.

Figure : Streamlit Input and Detection Output Preview (Image 2 with Enhancement Enabled)

A screenshot of a computer

AI-generated content may be incorrect.

Figure : Streamlit Detection Summary Table (Image 2)

A screenshot of a car on a road

AI-generated content may be incorrect.

Figure : Streamlit Detection Result Gallery Overview